UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
09/977,512	10/12/2001	David S. Allison	0007056-0197/P5940	3988

32615 7590 07/27/2006

OSHA LIANG L.L.P./SUN
1221 MCKINNEY, SUITE 2800
HOUSTON, TX 77010

EXAMINER	
VU, TUAN A	

ART UNIT	PAPER NUMBER
2193	

DATE MAILED: 07/27/2006

Please find below and/or attached an Office communication concerning this application or proceeding.

Office Action Summary	Application No.	Applicant(s)	
	09/977,512	ALLISON, DAVID S.	
	Examiner	Art Unit	
	Tuan A. Vu	2193	

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE 3 MONTH(S) OR THIRTY (30) DAYS, WHICHEVER IS LONGER, FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☒ Responsive to communication(s) filed on *26 April 2006*.

2a)☒ This action is **FINAL**. 2b)☐ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) *1-4 and 6-9* is/are pending in the application.

4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)☐ Claim(s) _____ is/are allowed.

6)☒ Claim(s) *1-4 and 6-9* is/are rejected.

7)☐ Claim(s) _____ is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)☐ The specification is objected to by the Examiner.

10)☐ The drawing(s) filed on _____ is/are: a)☐ accepted or b)☐ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11)☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

12)☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)☐ All b)☐ Some * c)☐ None of:

1.☐ Certified copies of the priority documents have been received.

2.☐ Certified copies of the priority documents have been received in Application No. _____.

3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

1) ☐ Notice of References Cited (PTO-892)
2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)
3) ☐ Information Disclosure Statement(s) (PTO-1449 or PTO/SB/08) Paper No(s)/Mail Date _____.
4) ☒ Interview Summary (PTO-413) Paper No(s)/Mail Date. *7/24/06*.
5) ☐ Notice of Informal Patent Application (PTO-152)
6) ☐ Other: _____.

DETAILED ACTION

1. This action is responsive to the Applicant's response filed 4/26/06.

As indicated in Applicant's response, claims 1 and 6 have been amended, with claims 5 and 10 previously canceled. Claims 1-4, 6-9 are pending in the office action.

Claim Rejections - 35 USC § 103

2. The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

> (a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

3. Claims 1-3, 6-8 are rejected under 35 U.S.C. 103(a) as being unpatentable over Yasumatsu, USPN: 5,579,518 (hereinafter Yasumatsu); in view of Chan et al., USPubN: 2005/0172302 (hereinafter Chan).

As per claim 1, Yasumatsu discloses a method for binding an object member at runtime comprising:

declaring said object member (e.g. *provisional type 12* - Fig. 2) in a program written in a dynamic typed programming language (see *provisional type 12* – Fig. 1-2), and running said program comprising:

determining whether said object member is used at runtime, and whether said object member is accessible (e.g. *routine lookup* – col. 7, lines 54-60; Fig. 3-4; col. 8, lines 18-26, 50-60); and

binding at runtime said object member to its reference if said object member is used and accessible (col. 8, line 61 to col. 9, line 54 – Note: push value into a stack after a dynamic type

checking reads on binding a object to its reference for a runtime use; and lookup method reads on whether object is accessible).

But Yasumatsu does not disclose runtime accessibility determination using an access control level; nor does Yasumatsu explicitly disclose declaring of the object written in the dynamically typed programming language so the declaring comprises assigning such access control level wherein a public member and private member have different access right. In a method to support runtime object resolution using statically provided information similar to the provisional type structure by Yasumatsu, Chan discloses providing package information for enabling runtime readjusting of class objects references with respect to reference to a function/method call – i.e. call invocation can be accessible - and discloses member access control wherein a Access violation can be detected (e.g. Fig. 2 – Note: *Illegal Access* violation reads on public and private member access incompatibility). In view of the object-oriented aspect of Yasumatsu's (see col. 1, lines 20-61) and the dynamic type binding so well-known in OO programming in which the inheritance as well as public and private access restrictions is known to be integral thereto, the declaring of some object so that access control is based on being a public or private member is disclosed. Hence, it would have been obvious for one of ordinary skill in the art at the time the invention was made to apply the dynamic type binding by Yasumatsu in combination with the lookup information in the context of call reference and runtime accessibility using the access control (Access violation at runtime due to an object being public or private) as mentioned above by Chan because problems encountered in known OO programming dynamic binding that come with polymorphism and parent-child hierarchical-type

access would be alleviated when the mapping of child/parent relationship for a call reference can be correctly redirected at runtime (see Chan para 0009 - 0017).

As per claim 2, Yasumatsu (in conjunction with Chan) discloses that object member is a class member of said dynamically typed programming language (e.g. *class to which... object belongs* - col. 5, lines 49-60).

As per claim 3, Yasumatsu (in conjunction with Chan) discloses that object member is a class method (e.g. Fig. 1, steps *13-14-15*).

As per claim 6, Yasumatsu discloses a computer useable medium having computer readable program code embodied therein configured to bind an object member at runtime, said computer program product comprising computer readable code configured therein to cause a computer to perform the steps:

to declare (object member...);

to determine (... object member is used); and whether said object member is accessible and

to bind (object member to its reference) exactly as recited in claim 1 above. Hence these above limitations are rejected with the corresponding rejection as set forth above therein.

But Yasumatsu does not disclose a runtime an access control level; nor does Yasumatsu explicitly disclose declaring of the object with such access control level wherein a public member and private member have different access rights. But these limitations have been addressed in claim 1 above.

As per claims 7-8, these claims correspond to claims 2-3, respectively, hence are rejected with the corresponding rejections as set forth therein.

4. Claims 4 and 9 are rejected under 35 U.S.C. 103(a) as being unpatentable over Yasumatsu, USPN: 5,579,518, and Chan et al., USPubN: 2005/0172302; in view of Admitted Prior Art (hereinafter APA).

As per claim 4, C++ virtual methods declared for runtime resolution of overridden functions or polymorphism-related issues has been known in the art (APA: specifications, BACKGROUND ART: p. 7-13) as well as SELF enhancement to approach late binding issues. Further, the use of message to dynamically resolve the reference to class method invocation has been mentioned in teachings by SmallTalk (base language for SELF) and Objective-C by Yasumatsu as improvements over known OO languages (see BACKGROUND; Fig. 1-2). Although Yasumatsu does not explicitly teach the object member is a virtual method, based on the APA and the various enhancements as above-mentioned, it would have been obvious for one of ordinary skill in the art at the time the invention was made to implement the dynamically typed process for runtime binding of class members as by the combination Yasumatsu/Chan so that virtual method members are also objects for such implementation because of the need to solve the drawbacks in OO languages (see APA- languages including C++) in regard to polymorphism or overriding of OO methods, such drawbacks also being at the root for the method by Yasumatsu as mentioned above.

As per claim 9, this claim incorporates the same rejection as set forth in claim 4, for including the same limitations therein.

Response to Arguments

5. Applicant's arguments with respect to claims 1-4, 6-9 as per the response filed 4/26/06 are not persuasive. Following are the Examiner's observations in regard thereto.

(A) Applicant has submitted that 'the object member may be declared as public or private, with access rights then depending, at run-time, on the declaration and the calling block's relationship to the object member' (Appl. Rmrks, pg. 5, 1st para). There is no such recital of a calling block in the claims; and the argument amounts to allegation that has apparently been based on knowledge garnered from outside the claim, hence not commensurate with Examiner's broad reasonable interpretation of the claim.

(B) Applicant has submitted that because Chan's access control is done via registration with a friendly object and based on such friend object, Chan's determining of access right is not the same as the invention wherein object members may be declared as public or private, so that access rights at runtime would depend from that declaration and the calling block's relationship to the object member (Appl. Rmrks, pg. 6, middle 2 paras). There is not sufficient description in the claim to enforce that the runtime determining of access rights is based on the declaration to preclude the access right by Chan to meet what has been recited as 'determining during run-time whether said object member is accessible using the access control level' because an Chan's illegalAccess violation is necessarily a exception dealing with access right of a class member; and the OO programming language as well known in the art (APA) allows such private or public characteristic as part of any member of a class when the class is declared. So Chan discloses determining whether that runtime access is violated based on the 'public' or 'private' right of a member in a class; and throws an exception at runtime. Chan has met the limitation which does not appear to be explicit in Yasumatsu's teaching. Runtime registration by Chan is not to be equated to declaration of a member because Chan's implements the determination at runtime as to whether or not a member would belong to a scope (a friend) for which the access is known;

i.e. this implements the determining step and based on the very nature of OO programming language allowing declaration of class member as set forth above, Chan is not necessarily brought in to meet this declaration limitation, because Chan is only purported to address a runtime determination as to whether an Access right (previously declared when the class has been defined) would have been violated.

(C) Applicant has submitted that because Chan teaches calling a function in a known class Chan teaches binding before determining access right (Appl. Rmrks, pg. 7, first para). The claim does not make it explicitly unequivocal that the binding is to be done with absolute absence of any further invocation to any other function; nor does the claim describe in sufficient terms the process of binding an object member as recited in the claim for Chan's approach for determining whether a runtime access has an exception to be completely improper. That is, the determining as effectuated by Chan's approach is done at runtime with some underlying techniques which amounts to determining at runtime if Access right has been violated. In terms of the interpretation derived from the way the claim is recited, there are not sufficient specifics in the claim to rebut that because Chan calls upon intermediate steps before some binding takes place, Chan's runtime binding is flawed. The claim should make it clear what the process of binding consists of, the context within which a member is to be determined as to its belonging to a right, i.e. which entity possesses that right in regard to the access of that very member when the runtime binding process is resolving possible relation between member and member's encompassing scope (container) type or privileges; and when the determining step occurs, there should be no intermediate invocation to any other function. The claim only teaches an interpretation in the line of determining whether said object member is used at runtime, and

whether said object member is accessible. Chan in view of Yasumatsu, in light of the rationale of USC 103 as set forth in the Office Action has met the above limitation, in view of broad reasonable interpretation based on well-known concept at the time the invention was made, or knowledge from Admitted Prior Art. Applicant's argument appear to have not considered the extent at which the language of the claim has not allow narrower interpretation; and it is the language of the claim that would support what Applicant deems to be his/her invention. Applicant's arguments fail to comply with 37 CFR 1.111(b) because they amount to a general allegation that the claims define a patentable invention without specifically pointing out how the language of the claims patentably distinguishes them from the references.

(E) Applicant has submitted that Chan's virtual method binding in C++ cannot be combined with Yasumatsu's late binding (Appl Rmrks, pg. 8, 1st and 2nd para). At the time the invention was made, Yasumatsu's object oriented object binding can be approached with the C++ by Chan, in light of knowledge of one skill in the art, based on the common endeavor by both Chan and Yasumatsu, to solve difficulties for some structure whose runtime references would remain incertain; and both Background of Yasumatsu and Chan relate to this shared concern. Hence, based on such common endeavor, at the time where C++ has its Private and Public member access right, one skill in the art would be motivated to combine this AccessRight Violation to support the late binding by Yasumatsu. The Applicant fails to show that if Yasumatsu's method can be modified to incorporate Chan's teaching, adverse effects would occur, i.e. by making Yasumatsu's approach such that Chan's programming language can be used to as to enable Public and Private access right declaration in a class member, Yasumatsu's invention would not succeed, and so, how precisely. In response to applicant's argument that Chan's C++ and

Yasumatsu's dynamic types programming would be nonanalogous art, it has been held that a prior art reference must either be in the field of applicant's endeavor or, if not, then be reasonably pertinent to the particular problem with which the applicant was concerned – that is, the analogous concern of resolving late referencing using either a candidate class (Yasumatsu) or a friend object (Chan) -- in order to be relied upon as a basis for rejection of the claimed invention. See In re Oetiker, 977 F.2d 1443, 24 USPQ2d 1443 (Fed. Cir.1992).

(F) Applicant has submitted that Chan makes use of existing features in C++ while the invention integrates access control in the dynamically typed languages (Appl. Rmrks, pg. 9, top). The claim is practically silent as to what 'dynamically typed programming' is all about, except for the declaring step, which includes assigning a Public and a Private member. These member access rights are integral to Chan's OO language, making the declaring step a known concept. Further, Chan is not purported to anticipate the invention but is intended to enhance the base art reference in a obviousness rationale. In response to applicant's arguments against the references individually, one cannot show nonobviousness by attacking references individually where the rejections are based on combinations of references. See In re Keller, 642 F.2d 413, 208 USPQ 871 (CCPA 1981); In re Merck & Co., 800 F.2d 1091, 231 USPQ 375 (Fed. Cir. 1986). Besides, there is not enough information in the language of the claim to enforce what Applicant calls 'integrating access control features into a dynamically typed programming language' when this very dynamic type aspect, which appears to be Applicant's main novel feature (in what way is the language dynamically typed?) is not put forth in more defined, and deliberate and sufficient terms or step actions description.

(G) The invention as recited presents further deficiencies. The binding at runtime is prone to not actualize at all; and all of Applicant's above arguments would be based on an invention that would or will not realize.

Specifically, claim 1 (or claim 6) recites "running said program... comprising... determining whether said object ...is used...determining whether said object ... is accessible... binding at runtime ... if said object ... is used and if said object ... is accessible".

As recited, the two steps of *determining* and one of *binding* amount to what is recited as 'running said program' after the step of declaring. The binding is dependent on two determining steps which are again contingent or dependent upon 2 more conditions such as in the *if* or *whether*; that is, (i) if an object member is used and (ii) if said very member is accessible in order for the binding step to take place, not otherwise. For one skill in the art, these steps are not provided with sufficient alternatives in terms of what would be done once the above conditions were not met. Absent any action as from such an alternative, one skill in the art would not be reasonably apprised on any action that would otherwise take place in order for the claimed steps to realize into a useful, concrete and tangible result as statutorily, i.e. the claimed invention being a strong candidate for a USC 101 type of rejection.

For the above reasons, the claims will stand as rejected in the Office Action.

Conclusion

6. **THIS ACTION IS MADE FINAL.** Applicant is reminded of the extension of time policy as set forth in 37 CFR 1.136(a).

A shortened statutory period for reply to this final action is set to expire THREE MONTHS from the mailing date of this action. In the event a first reply is filed within TWO

MONTHS of the mailing date of this final action and the advisory action is not mailed until after the end of the THREE-MONTH shortened statutory period, then the shortened statutory period will expire on the date the advisory action is mailed, and any extension fee pursuant to 37 CFR 1.136(a) will be calculated from the mailing date of the advisory action. In no event, however, will the statutory period for reply expire later than SIX MONTHS from the mailing date of this final action.

Any inquiry concerning this communication or earlier communications from the examiner should be directed to Tuan A Vu whose telephone number is (272) 272-3735. The examiner can normally be reached on 8AM-4:30PM/Mon-Fri.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Kakali Chaki can be reached on (571)272-3719.

The fax phone number for the organization where this application or proceeding is assigned is (571) 273-3735 (for non-official correspondence – please consult Examiner before using) or 571-273-8300 (for official correspondence) or redirected to customer service at 571-272-3609.

Any inquiry of a general nature or relating to the status of this application should be directed to the TC 2100 Group receptionist: 571-272-2100.

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see http://pair-direct.uspto.gov. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free).

Tuan A Vu
VAT
January 12, 2005

KAKALI CHAKI
SUPERVISORY PATENT EXAMINER
TECHNOLOGY CENTER